02

Agregando la libreria auth0-jwt

Transcripción

[00:00] ¿Qué tal? Bienvenidos a su cuarta clase de su curso de Spring Security. Esta vez vamos a comenzar viendo un poco sobre en qué parte estamos en el proceso de autenticación.

[00:11] Si recuerdan bien, hasta la clase anterior vamos un rato a ver el código, vimos ya lo que es las clases propias de Spring, por ejemplo aquí en el Security, para crear nuestra configuración de seguridad y sobrescribir la configuración por defecto que Spring nos trae usar nuestro AutenticacionService.

[00:31] Para cargar nuestro usuario desde nuestra base de datos y a través de nuestro controller en nuestro AutenticacionController cargar ese usuario, generar el token y revolver ok en nuestro ResponseEntity.

[00:47] Vimos que flujo ya funciona end to end, digamos, de inicio a fin, porque ya desde el controller de nuestro API ya recibimos el DTO para los datos de autenticación y ya retornamos el usuario desde nuestra base de datos que lo hemos guardado aquí en nuestra tabla de usuarios.

[01:10] Vemos que aquí está el hash existente y todo está bien hasta aquí. A nivel de nuestro diagrama, por ejemplo, ya vimos que como primer paso nuestro usuario clave son enviados en el body, a nuestro endpoint de login, nuestro API Rest. Nuestro API Rest hace un select dentro de la base de datos y ahora, lo que él tiene que hacer es generar el JSON Web Token y devolver el JSON Web Token.

[01:36] En esta clase vamos a centrarnos específicamente en estos dos últimos pasos. Cómo genera ese JSON Web Token y cómo devuelvo ese JSON Web Token. Y lo que muchos de ustedes se deben estar preguntando ¿qué es un JSON Web Token? Bueno, vamos a comenzar esto abriendo Google y vamos a entrar a jwt.io.

[02:00] Acepto las cookies requeridas. Listo. Y esta es la página oficial de JWT JSON Web Token, como lo quieran llamar. JSON Web Token es un estándar para definición de tokens en el cual, por ejemplo el token generado es generado con un algoritmo de encriptación, por ejemplo, tenemos varios algoritmos irritación, por ejemplo aquí vemos que estaba usando un HS256, 256 bits, del tipo JWT.

[02:34] Vemos que tenemos una sección de payload, que tiene ciertos datos como un usuario, un nombre, un sub o un issue add, que es básicamente a qué hora fue firmado y una parte que es la firma de este token.

[02:50] Podemos aquí en la parte que está aquí en encoded también, por ejemplo si tú tienes algún token JWT y quieres validarlo, quieres decodificarlo, bueno, en esta página puedes copiarlo o pegarlo, escoger el algoritmo de encriptación que hayas usado y en efecto, puedes ver la información que está dentro de ese token.

[03:09] Como pueden ver, a alto nivel esto es un estándar o estándar de la industria para lo que son tokens de autenticación. Ahora si yo quiero usar este estándar, mi JSON Web Token, yo necesito ahora una librería especial para crear mi JSON Web Token aquí en el proyecto, para eso voy a venir aquí a ver librerías de JWT.

[03:33] Y cómo pueden ver aquí, por ejemplo, yo tengo a librerías para .NET, tengo librerías, tengo muchas para .NET, para Ada, distintos lenguajes de programación. Como no quiero recorrer toda lista lo que voy a hacer es un filtro y voy a escoger Java. Bajo un poco y aquí está Java.

[03:55] Filtro lo que tengo para Java va y vemos que aquí en la primera opción tengo la librería de Auth0. Auth0 es una compañía que se especializa en proveer plataformas de autenticación de validación de la identidad del usuario. Fue comprada Okta hace poco, pero hasta hace un poco menos de un año era una compañía independiente, una compañía de Argentina muy respetada en toda la industria.

[04:23] Ellos hacen colaboraciones siempre para el mundo open source, para la comunidad. Una de las colaboraciones es una librería para generar JSON Web Tokens. Entonces vamos a utilizar este de aquí, es la primera opción, al día de hoy estamos febrero del 2023.

[04:41] Para descargar la librería donde yo voy a ir es a la fuente del repositorio, es donde voy a dar clic en view repo y me va a llevar al repositorio en GitHub. ¿Qué va a pasar aquí en GitHub? Aquí siempre en todo repositorio tienes la nota del readme. Entonces en el readme tú vas a ver aquí que hay una sección de getting started que te va a dar alguna descripción sobre compatibilidad de Java con esta librería.

[05:09] Y también la información de cómo instalar esto en tu proyecto actual. Si estás usando Maven, te da el escape que estás usando de Maven. Si estás usando Gradle, te va a dar entonces la pieza de código que necesitas para implementarlo en Gradle.

[05:27] También te va a dar información de cómo crear un JWT, cómo verificar un JWT. No se preocupen, vamos a verlo poco a poco. Nuestro goal por ahora es integrar esta librería en nuestro proyecto. Entonces vamos a ver aquí el último release es de la semana pasada, pero normalmente los releases más recientes a veces presentan un tipo de bugs que son solucionados en los siguientes.

[05:56] Para tener una versión más estable yo voy usar una versión menor que sería la 4.2.0. ¿Cómo voy a seleccionar esto? Voy a copiar acá, regreso a mi

proyecto y entró a mi pom.xml. Nuevamente así como adicioné la dependencia de la librería de Spring Boot doy enter y agrego la de Auth0 Java JWT, pero con la versión .2.

[06:22] Por favor, usen esta versión ustedes también porque ya saben que entre versiones pueden haber algunos cambios y quizás los ejemplos que yo use aquí ya no funcionen en la nueva versión. Vale la pena recalcar que en esta librería como ya les comenté antes no es propia del ecosistema de Spring. Al igual que el Longbok, por ejemplo esta es de un proveedor independiente en este caso Auth0.

[06:47] Mi servidor está detenido. Voy aquí a Maven y voy a descargar esa dependencia nuevamente. Vemos que está resolviendo mis dependencias aquí, ya terminó y ahora miren cómo parece aquí, com.auth0:java-jwt:4.2.0. Ya con esto ya estamos listos para comenzar a generar nuestros tokens en el formato JWT. Eso lo vamos a ver en el siguiente video.